BAB I
PENDAHULUAN

1.1 Latar Belakang

Deteksi mata mengantuk menghadapi tantangan tambahan ketika objek yang dianalisis menggunakan kacamata. YOLO (You Only Look Once), sebuah metode arsitektur deteksi objek berbasis deep learning, telah terbukti efektif dalam berbagai aplikasi, namun penggunaannya untuk mendeteksi mata mengantuk dalam kondisi yang beragam, seperti menggunakan kacamata atau tidak, masih memerlukan penelitian lebih lanjut. YOLO dikenal dengan kemampuannya untuk melakukan deteksi objek secara real-time dengan akurasi yang tinggi, tetapi keberhasilannya sangat bergantung pada kualitas data dan model pelatihan yang digunakan. Permasalahan yang dihadapi termasuk bagaimana mengatasi refleksi cahaya pada lensa kacamata yang dapat mengganggu proses deteksi, serta variabilitas bentuk dan ukuran kacamata yang dapat mempengaruhi hasil deteksi. Oleh karena itu, penelitian ini berfokus pada pengaruh penggunaan kacamata terhadap efektivitas deteksi mata mengantuk menggunakan arsitektur YOLO, dengan tujuan untuk mengembangkan model yang akurat dalam berbagai kondisi [1].

1.2 Rumusan Masalah

1. Bagaimana cara mengimplementasikan arsitektur YOLOv8 dalam sistem deteksi mata kantuk secara efisien?
2. Bagaimana performa deteksi mata kantuk menggunakan arsitektur YOLOv8 dalam berbagai kondisi pencahayaan dan latar belakang visual?
3. apa perbedaan kinerja YOLO 8 pada data subyek berkacamata dan tidak berkacamata?

1.3 Batasan Masalah

1. Fokus pada penerapan arsitektur YOLOv8 untuk mendeteksi tanda-tanda mata kantuk.
2. Penilaian performa deteksi mata kantuk di bawah kondisi pencahayaan dan latar belakang visual yang berbeda.

3. Analisis perbedaan kinerja YOLOv8 pada subyek yang memakai kacamata dibandingkan dengan yang tidak memakai kacamata.

1.4 Tujuan Penelitian

1. Mengembangkan dan mengimplementasikan arsitektur YOLOv8 untuk mendeteksi tanda-tanda mata kantuk secara efisien dan akurat.
2. Mengevaluasi performa deteksi mata kantuk menggunakan arsitektur YOLOv8 di bawah berbagai kondisi pencahayaan dan latar belakang visual untuk memastikan keandalan sistem dalam situasi nyata.
3. Menganalisis perbedaan kinerja YOLOv8 dalam mendeteksi mata kantuk pada subyek yang memakai kacamata dan yang tidak memakai kacamata, untuk memahami bagaimana aksesori wajah dapat mempengaruhi efektivitas deteksi.

1.5 Manfaat Penelitian

1. Menyediakan metode yang efisien untuk mengimplementasikan YOLOv8 dalam sistem deteksi mata kantuk, yang dapat digunakan dalam aplikasi keamanan dan kesehatan, seperti sistem pemantauan pengemudi untuk mencegah kecelakaan
2. Memahami performa YOLOv8 dalam berbagai kondisi pencahayaan dan latar belakang visual membantu meningkatkan robusta sistem deteksi mata kantuk, sehingga dapat berfungsi dengan baik di berbagai lingkungan nyata.
3. Mengetahui perbedaan kinerja deteksi pada subyek berkacamata dan tidak berkacamata memungkinkan penyesuaian model untuk mengatasi kendala yang mungkin timbul akibat aksesori wajah, sehingga memperbaiki akurasi dan keandalan sistem.

1.6 Sistematika Penulisan

Sistematika penulisan tugas akhir sebagai berikut :

BAB I PENDAHULUAN

Memberikan penjelasan terkait latar belakang permasalahan, rumusan masalah, Batasan permasalahan, tujuan penelitian, manfaat penelitian, dan sistematika penelitian

BAB II TINJAUAN PUSTKA

Menguraikan tentang konsep dan penjelasan metode yang digunakan. Selain itu terdapat hasil hasil penelitian yang telah dilakukan sebelumnya oleh peneliti lain yang terkait dengan penelitian yang dilakukan.

BAB III METODE PENELITIAN

Menjelaskan terkait alur penelitian, Teknik yang digunakan serta data yang akan dikaji.

BAB IV HASIL DAN PEMBAHASAN

Menguraikan tentang data yang diperoleh yang ditampilkan dalam bentuk table dan gambar serta cara menganalisa data tersebut. Pengolahan data termasuk analisis yang dilakukan terhadap data yang diperoleh.

BAB V PENUTUP

Memaparkan pembahasan berupa hasil dar penelitian dan saran untuk kemajuan penelitian ini.

BAB II
TINJAUAN PUSTAKA

2.1 Studi Literatur

Pada penelitian pertama yaitu dengan judul Performa Model YOLOv8 untuk Deteksi Kondisi Mengantuk yang dibuat oleh Edmund Ucok Armin, Anggun Purnama Edra, Fakhri Ikhwanul Alifin, Ikhwanussafa Sadidan, Indri Purwita Sary, Ulinnuha Latifa yang dipublikasikan pada tahun 2023. Peneletian ini yaitu untuk mendeteksi kondisi pengemudi saat mengantuk menggunakan algoritma YOLOv8. Penelitian ini menunjukan hasil kinerja mencapai nilai mAP sebesar 0.96092 selama proses pelatihan dengan memanfaatkan 2996 citra dari dataset sekunder [2].

Pada penelitian kedua ini berjudul Driver Drownsiness Monitoring System Using YOLOv8 yang dibuat oleh P.Vijay Bhaskar Reddy, S.Sai Rachana yang di publikasikan pada tahun 2023 ini yaitu mendeteksi pengemudi mengantuk menggunakan algoritma YOLOv8. Penelitian ini menunjukan hasil diperoleh tingkat accuracy sebesar 91,1%. Model ini cepat dan akurat dibandingkan model berbasis algoritma KNN lainnya mendeteksi objek dalam satu tembakan daripada deteksi dua tahap [1].

Pada penelitian ketiga ini berjudul Klasifikasi Penyakit Mata Berdasarkan Citra Fundus Menggunakan YOLOv8 yang dibuat oleh P Muhammad Nur Ihsan Muhlashin, Arnisa Stefanie yang di publikasikan pada tahun 2023 ini yaitu mengklasifikasikan penyakit mata berdasarkan citra fundus menggunakan algoritma YOLOv8. Penelitian ini mendapatkan model yang dihasilkan menunjukkan nilai accuracy sebesar 92%, precision sebesar 91%, recall sebesar 92%, F1-score sebesar 91% [3].

Pada penelitian keempat ini berjudul YOLOv8 Peningkatan Algoritma Untuk Deteksi Pemakaian Masker Wajah yang dibuat oleh Yanto, Faruq Aziz, Irmawati yang di publikasikan pada tahun 2023 ini yaitu mendeteksi pemakaian masker untuk wajah menggunakan algoritma YOLOv8. Penelitian ini mendapatkan model yang dihasilkan menunjukkan akurasi kelas badmask sebesar 94%, mask

sebesar 97% dan nomask sebesar 95%. Nilai F1-Confidence, Precision, dan Recall semua kelasnya juga tinggi, yaitu masing-masing sebesar 0.94, 0.96, dan 0.97% [4].

Pada penelitian kelima ini berjudul Aplikasi Warning Alert Pendeteksi Kelelahan Ekspresi Wajah Pada Pengemudi Secara Real-Time Menggunakan Metode You Only Look Once Berbasis Website yang dibuat oleh Hafidh Ahmad Fauzan1, Ari Kurniawan yang di publikasikan pada tahun 2023 ini yaitu membuat aplikasi warning alert untuk mendeteksi kelelahan ekspresi wajah pada engemudi secara real time menggunakan metode YOLO berbasis website. Penelitian ini mendapatkan model yang dihasilkan menunjukkan model mampu menghasilkan nilai akhir mAP sebesar 1.000000 atau 100.00%, nilai akhir Precision sebesar 0.94, nilai akhir Recall sebesar 1.00 dan nilai akhir F1-Score sebesar 0.97 hanya dengan 20 data saja [5].

Pada penelitian keenam ini berjudul Perancangan Alat Identifikasi Wajah Dengan Algoritma You Only Look Once (YOLO) Untuk Presensi Mahasiswa yang dibuat oleh Irma Salamah, M. Redho Ali Said, Sopian Soim yang di publikasikan pada tahun 2023 ini yaitu untuk merancang alat identifikasi wajah dengan algoritma YOLO untuk membuat presensi pada mahasiswa. Penelitian ini mendapatkan model hasil pengujian akurasi sistem dengan sampel 20 sampel mahasiswa yang dilakukan dalam proses pengenalan wajah melalui absensi dengan rata-rata akurasi 0,9793. Akurasi maksimum yang didapatkan 0,995 dan akurasi minimum yang didapatkan adalah 0,995 [6].

Tabel 2.1 Keaslian Penelitian

No	Judul penelitian	Nama Penulis	Tahun Publikasi	Hasil Penelitian	Perbandingan Penelitian	Ref
1	Performa Model YOLOv8 untuk Deteksi Kondisi Mengantuk pada pengendara mobil	Edmund Ucok Armin, Anggun Purnama Edra, Fakhri Ikhwanul Alifin, Ikhwanussafa Sadidan, Indri Purwita Sary, Ulinnuha Latifa	2023	mencapai nilai mAP sebesar 0.96092 selama proses pelatihan dengan memanfaatkan 2996 citra dari dataset sekunder, .	Mendekteksi objek orang dan mendeteksi objek bergerak	[2]
2	DRIVER DROWSINESS MONITORING SYSTEM USING YOLOV8	P.Vijay Bhaskar Reddy, S.Sai Rachana	2023	diperoleh tingkat accuracy sebesar 91,1%. Model ini cepat dan akurat dibandingkan model berbasis algoritma KNN lainnya mendeteksi objek dalam satu tembakan daripada deteksi dua tahap.	Mendekteksi objek orang dan mendeteksi objek bergerak	[1]

Tabel 2.1 Keaslian Penelitian

3	KLASIFIKASI PENYAKIT MATA BERDASARKAN CITRA FUNDUS MENGGUNAKAN YOLO V8	Muhammad Nur Ihsan Muhlashin, Arnisa Stefanie	2023	Model yang dihasilkan menunjukkan nilai accuracy sebesar 92%, precision sebesar 91%, recall sebesar 92%, F1-score sebesar 91%.	Mendekteksi objek mata dan mendeteksi objek bergerak	[3]
4	YOLO-V8 PENINGKATAN ALGORITMA UNTUK DETEKSI PEMAKAIAN MASKER WAJAH	Yanto, Faruq Aziz, Irmawati	2023	akurasi kelas badmask sebesar 94%, mask sebesar 97% dan nomask sebesar 95%. Nilai F1-Confidence, Precision, dan Recall semua kelasnya juga tinggi, yaitu masing-masing sebesar 0.94, 0.96, dan 0.978.	Mendekteksi objek wajah dan mendeteksi objek bergerak	[4]

Tabel 2.1 Keaslian Penelitian

5	Aplikasi Warning Alert Pendeteksi Kelelahan Ekspresi Wajah Pada Pengemudi Secara Real-Time Menggunakan Metode You Only Look Once Berbasis Website	Hafidh Ahmad Fauzan1, Ari Kurniawan	2023	Model mampu menghasilkan nilai akhir mAP sebesar 1.000000 atau 100,00%, nilai akhir Precision sebesar 0.94, nilai akhir Recall sebesar 1.00 dan nilai akhir F1-Score sebesar 0.97 hanya dengan 20 data saja.	Mendekteksi objek wajah dan mendeteksi objek bergerak	[5]

Tabel 2.1 Keaslian Penelitian

6	Perancangan Alat Identifikasi Wajah Dengan Algoritma You Only Look Once (YOLO) Untuk Presensi Mahasiswa	Irma Salamah, M. Redho Ali Said, Sopian Soim	2022	Hasil pengujian akurasi sistem dengan sampel 20 sampel mahasiswa yang dilakukan dalam proses pengenalan wajah melalui absensi dengan rata-rata akurasi 0,9793. Akurasi maksimum yang didapatkan 0,995 dan akurasi minimum yang didapatkan adalah 0,995	Mendekteksi objek orang dan mendeteksi objek bergerak	[7]

2.2 Dasar Teori

2.2.1 Kantuk

Kantuk atau 'ngantuk' adalah kondisi ketika seseorang merasa ingin tidur. Kondisi ini biasa terjadi pada malam hari atau kadang di siang hari. Kantuk merupakan hal yang wajar, tetapi perlu diatasi jika terjadi secara berlebihan, mengganggu aktivitas, atau menurunkan produktivitas. Kantuk umumnya muncul karena kurang tidur. Meskipun terlihat sederhana, kantuk bisa memicu timbulnya berbagai masalah, seperti mengganggu prestasi di sekolah atau produktivitas di kantor, memengaruhi emosi, atau bahkan menyebabkan kecelakaan, baik di jalan raya maupun di lingkungan kerja. Rasa 'ngantuk' yang tidak normal bisa merupakan tanda dari penyakit, seperti *sleep apnea*, narkolepsi, insomnia, *restless leg syndrome*, depresi, rasa kecemasan, atau diabetes. Artikel ini akan membahas kantuk yang tidak normal [8].

2.2.2 Artificial Inteligent

Apa itu AI? AI adalah Kecerdasan Buatan, seperti kepanjangan AI yaitu Artificial Intelligence, AI merupakan teknologi yang dirancang untuk membuat sistem komputer mampu meniru kemampuan intelektual manusia. AI memungkinkan komputer untuk belajar dari pengalaman, mengidentifikasi pola, membuat keputusan, dan menyelesaikan tugas-tugas kompleks dengan cepat dan efisien [6].

2.2.3 Mchine Learning

Machine learning merupakan salah satu cabang dari kecerdasan buatan atau artificial intelligence (AI) dan ilmu komputer. Machine learning berfokus pada penggunaan data dan algoritma untuk membuat solusi yang mirip dengan cara manusia belajar sehingga dapat memperbaiki diri secara bertahap. Dalam pengembangan data science, machine learning adalah komponen yang sangat penting. Melalui metode statistika, algoritma pada machine learning dilatih untuk membuat klasifikasi atau prediksi, sehingga mampu memberikan insight utama

selama proses pengolahan data. Insight ini akan berpengaruh besar terhadap penentuan arah sebuah penelitian atau bisnis [9].

2.2.4 Deep Learning

Deep learning, atau pembelajaran mendalam, merupakan paradigma dalam Artificial Intelligence yang mengadopsi pendekatan mirip otak manusia untuk pemrosesan informasi. Inti dari konsep ini terletak pada penggunaan jaringan saraf buatan dengan banyak lapisan *(deep neural networks)*. Inspirasi utamanya berasal dari cara otak manusia mengenali pola, belajar dari pengalaman, dan membuat keputusan. Banyaknya lapisan tersebut memungkinkan teknologi ini untuk menangani tugas-tugas yang kompleks, yang melibatkan pemahaman konteks dan pengenalan pola non-linear dalam data. Ini membuatnya sangat efektif dalam berbagai aplikasi seperti pengenalan wajah, identifikasi objek, dan pemrosesan bahasa alami, menjadikannya salah satu pilar utama dalam evolusi Artificial Intelligence [10].

2.2.5 Computer Vision

Computer vision adalah subbidang dari kecerdasan buatan *(Artificial Intelligence – AI)* yang berfokus pada pengembangan sistem komputer yang mampu "melihat" dan memahami dunia fisik dengan cara yang serupa dengan manusia. Tujuan utama dari computer vision adalah mengajarkan komputer untuk memahami dan menginterpretasi data visual, seperti gambar dan video. Computer vision dalam AI didedikasikan untuk pengembangan sistem otomatis yang dapat menafsirkan data visual (seperti foto atau gambar bergerak) dengan cara yang sama seperti manusia. Ide di balik computer vision adalah untuk menginstruksikan komputer dalam menafsirkan dan memahami gambar berdasarkan piksel demi piksel. Ini adalah dasar dari bidang computer vision. Dari sisi teknis, komputer akan berupaya mengekstrak data visual, mengelolanya, dan menganalisis hasilnya menggunakan program perangkat lunak yang canggih [11].

2.2.6 Yolo (You Only Look Once)

YOLO *(You Only Live Once)* adalah model visi komputer populer yang mampu melakukannya mendeteksi dan mensegmentasi objek dalam gambar. Model tersebut telah melalui beberapa kali pembaruan di masa lalu, dengan YOLOv8 menandai versi ke-8. Saat ini, YOLOv8 mengembangkan kemampuan versi sebelumnya dengan memperkenalkan fitur dan peningkatan baru yang canggih. Hal ini memungkinkan deteksi objek secara real-time dalam data gambar dan video dengan peningkatan akurasi dan presisi berikut dibawah ini yaitu Gambar 2.1 contoh deteksi menggunakan YOLO [12].

Gambar 2.1 Deteksi Objek Sumber : [12].

Akurasi YOLOv8 yang tak tertandingi dan kecepatan tinggi membuat model visi komputer menonjol dari versi sebelumnya. Ini merupakan pencapaian penting karena objek kini dapat dideteksi secara real-time tanpa penundaan, tidak seperti versi sebelumnya. Model YOLOv8 dapat disesuaikan agar sesuai dengan kasus penggunaan tertentu atau dilatih sepenuhnya dari awal untuk membuat model khusus. Rincian lebih lanjut tentang prosedur pelatihan dapat ditemukan di dokumentasi resmi [12]. YOLOv8 bekerja dengan membagi gambar input menjadi grid, di mana setiap sel pada grid bertanggung jawab untuk memprediksi bounding box dan kelas objek yang mungkin ada di dalam area tersebut. Proses prediksi ini dilakukan dalam satu tahap, yang memungkinkan deteksi objek

dilakukan secara cepat dengan akurasi tinggi. Arsitektur YOLOv8 mencakup fitur seperti backbone convolutional neural network (CNN) yang ditingkatkan, kepala deteksi (detection head) yang lebih baik, serta penggunaan loss function yang lebih optimal untuk meningkatkan presisi dan recall dalam deteksi objek [13].

2.2.7 Roboflow

RoboFlow adalah tolok ukur multidomain yang beragam dari kumpulan data yang terdiri dari 100 kumpulan data, 7 domain citra, 224.714 gambar, dan 805 label kelas. Itu berasal dari lebih dari 90.000 kumpulan data publik dan 60 juta gambar publik yang secara aktif disusun dan diberi label oleh praktisi visi komputer di aplikasi web terbuka RoboFlow Universe [14].

Gambar 2.2 Logo Roboflow

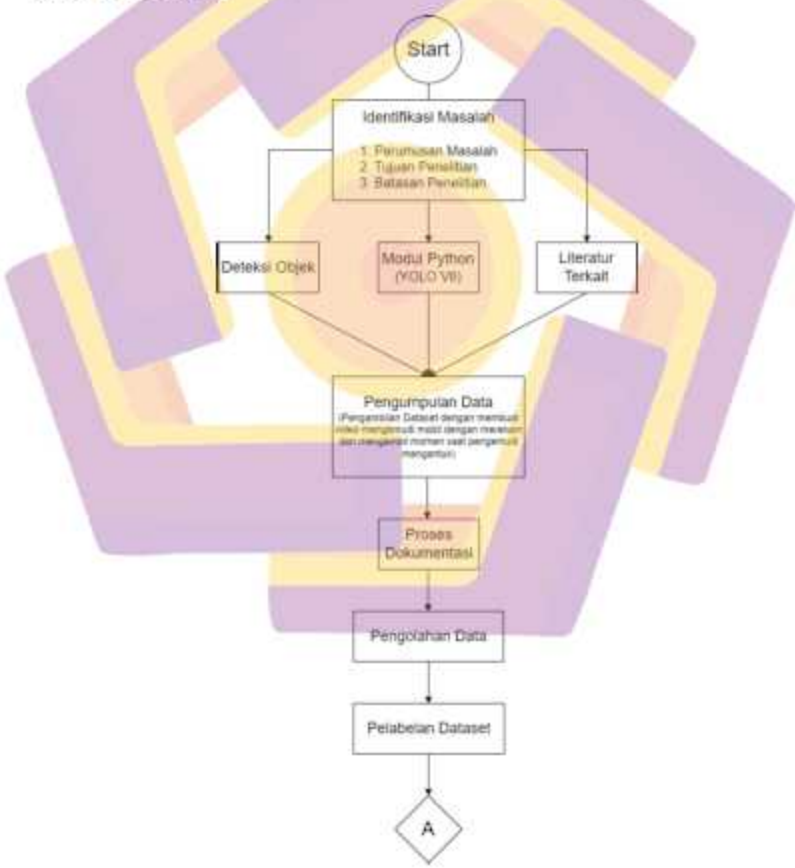
Tujuan merilis RoboFlow adalah untuk menyediakan para peneliti dengan tolok ukur untuk menguji generalisasi model deteksi objek mereka dengan data kehidupan nyata . Selain itu, RoboFlow adalah bahasa pemrograman visual berbasis aliran yang dirancang untuk memungkinkan pengguna akhir untuk memprogram tugas manipulasi seluler yang dapat digeneralisasi. Ini memastikan implementasi prosedur program tingkat rendah yang kuat pada manipulator seluler sambil membatasi pemrograman tingkat tinggi untuk menghindari kesalahan pengguna. RoboFlow telah diimplementasikan pada manipulator seluler PR2 dan menunjukkan sifat generalisasi dan penanganan kesalahannya pada tugas manipulasi seluler sehari-hari di lingkungan manusia [14].

BAB III
METODE PENELITIAN

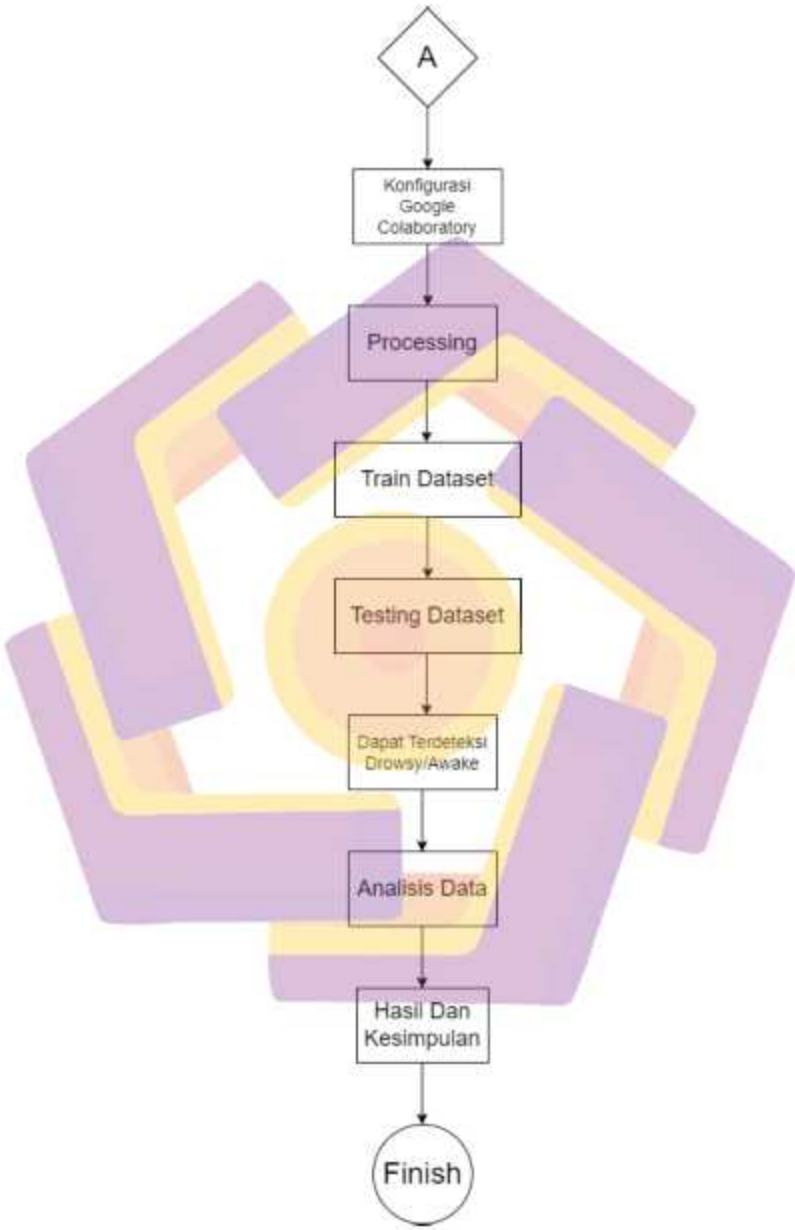
3.1 Objek Penelitian

Objek penelitian pada penelitian ini adalah gambar pengemudi yang mengantuk *(drowsy)* dan terjaga *(awake)* dengan menggunakan kacamat atau tidak untuk dilatih pada sistem agar dapat digunakan untuk mendeteksi kantuk pada pengemudi.

3.2 Alur Penelitian



Gambar 3.1 Alur Penelitian



Gambar 3.1 Alur Penelitian

3.4 Identifikasi Masalah

Pada tahap identifikiasi ini menjadi tahap awal untuk mengenalkan permasalahan yang akan diangkat dan ditentukan dari apa tujuan dari penelitian yang dilakukan tersebut.

3.5 Pengumpulan Data

Pada pengumpulan data ini dilakukan dengan mengumpulkan data yang sudah kita buat menggunakan aplikasi *Roboflow*.

3.6 Pengolahan Data

Pada proses ini kita melakukan *pre-processing* yaitu pelabelan data dengan melakukan anotasi dari foto yang sudah di convert deengan aplikasi roboflow. Setelah melakukan tahap *pre-processing* dilakukan *konfigurasi Google Colaboratory* untuk melakukan training data dan testing data.

3.7 Analisis Data

Proses dilakukan apabila model yang sudah dibuat dapat mendeteksi mengantuk*(drowsy)* atau terjaga*(awake)* apabila belum terdeteksi bisa melakukan tahap pre-processing data ulang, apabila mendeteksi kurang dari 90%, maka akan dilakukan konfigurasi ulang, apabila berhasil mendeteksi lebih dari 90% maka akan dilakukan analisis data sesuai data yang diolah.

3.8 Kesimpulan dan Saran

Tahap terakhir ini kita ditutup dengan menuliskan kesimpulan dari proses yang dilukan dengan menjawab rumusan masalah, dan menuliskan saran yang dapat menjadi refrensi untuk penelitian selanjutya.

3.3 Alat dan Bahan

Dalam penelitian ini Langkah pertama yaitu kita menyiapkan dataset terlebih dahulu. Yaitu sebuah video seperti Gambar 3.2 dan 3.3 di bawah ini.

Gambar 3.2 Video Bahan Dataset Siang Hari

Gambar 3.3 Video Dataset Malam Hari

Tabel 3.1 Klasifikasi Dataset Siang Hari

Class Name	Deskripsi	Total Gambar
Tidak Mengantuk	Gambar Mata Terbuka	269
Mengantuk	Gambar Mata Tertutup	922
Tidak Menguap	Gambar Mulut Terbuka	55
Menguap	Gambar Mulut Tertutup	534

Tabel 3.2 Klasifikasi Dataset Malam Hari

Class Name	Deskripsi	Total Gambar
Tidak Mengantuk	Gambar Mata Terbuka	307
Mengantuk	Gambar Mata Tertutup	652
Tidak Menguap	Gambar Mulut Terbuka	58
Menguap	Gambar Mulut Tertutup	405

Tabel 3.3 Ragam Gambar Pada Dataset

Gambar	Keterangan
	Mata Terbuka
	Mata Tertutup

	
 	Mulut Terbuka
 	Mulut Tertutup

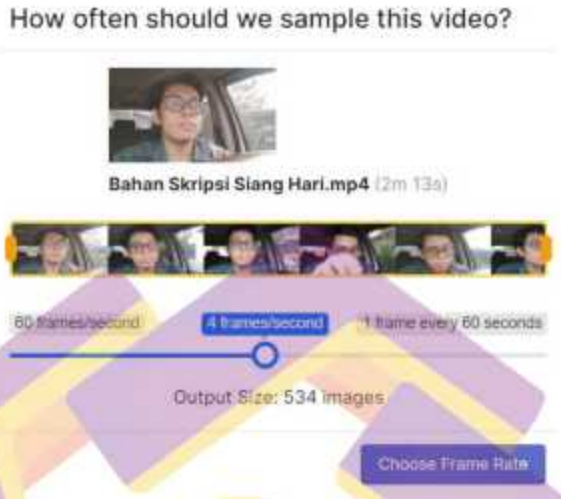
BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengumpulan Dataset

Gambar 4.1 Proses Pembauatan Dataset

Pada Gambar 4.1 diatas tahap pertama ini membuat dataset dengan merekam video saat sedang menyetir kendaraan mobil dan membuat simulasi mengantuk dan terjaga. Pada pembuatan video ini mengambil dalam 2 keadaan yaitu siang dan malam agar mengetahui perbandingan yang di hasilkan nanti.

Setelah membuat dataset selanjutnya pergi ke *roboflow*. *Roboflow* adalah kerangka kerja pengembang computer vision untuk pengumpulan data yang lebih baik hingga prapemrosesan, dan teknik pelatihan model. Dengan menggunakan *Roboflow* dapat membagikan dataset sekaligus memproses dataset tersebut melakukan *annotate* atau menandai objek yang akan di deteksi menggunakan *bounding* atau labeling objek yang akan di deteksi. Selanjutnya kita mengupload video ke roboflow kita membuat video menjadi gambar yaitu dengan mengconvert video menjadi gambar sebanyak 534 gambar.



Gambar 4.2 Proses *Convert* Video Menjadi Foto.

Pada Gambar 4.2 diatas Setelah proses *convert* seleasi maka didaptkan gambar sebanyak 534 seperti Gambar 4.3 di bawah ini.

Gambar 4.3 Bahan Dataset

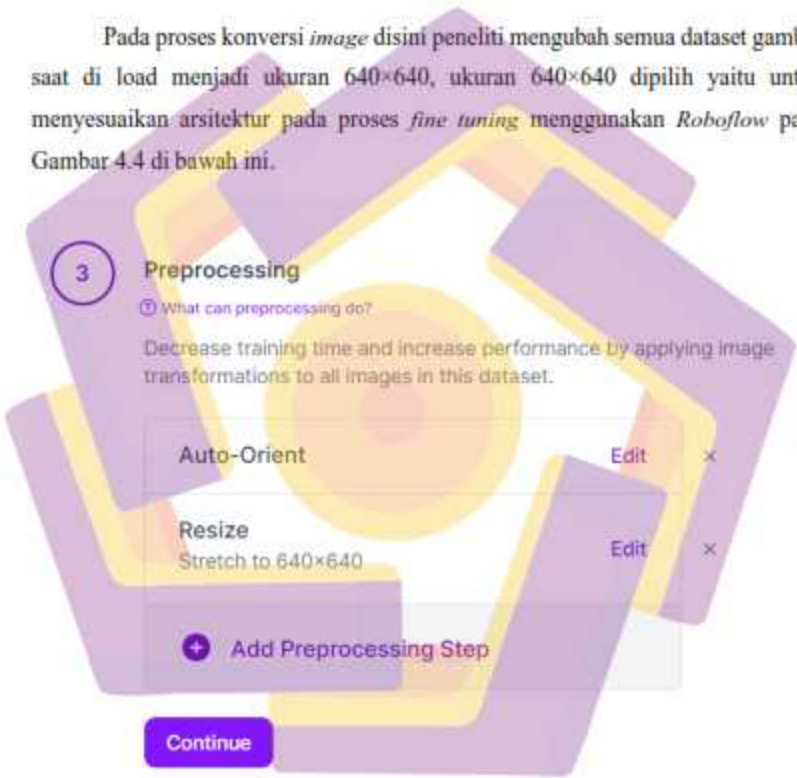
Jika sudah terupload, bisa melanjutkan ke tahap selanjutnya yaitu melakukan *labelling* pada setiap foto untuk menghasilkan berbagai *class* agar bisa dikategorikan sesuai kebutuhan yang dibutuhkan.

4.2 Preprocessing Data

Setelah terkumpulnya dan teruploadbahan untuk membuat dataset, selanjutnya dilakukan *preprocessing* citra dengan beberapa tahapan sebagai berikut.

4.2.1 Image Conversion

Pada proses konversi *image* disini peneliti mengubah semua dataset gambar saat di load menjadi ukuran 640×640, ukuran 640×640 dipilih yaitu untuk menyesuaikan arsitektur pada proses *fine tuning* menggunakan *Roboflow* pada Gambar 4.4 di bawah ini.



Gambar 4.4 Proses Image Conversion

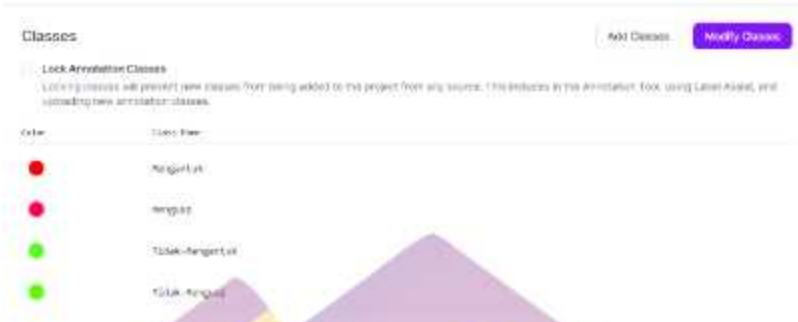
4.2.2 Labelling

Proses labeling dalam machine learning, khususnya di bidang computer vision dan pengenalan pola, adalah langkah penting untuk memberi label atau anotasi pada data mentah. Labeling melibatkan menandai data, seperti gambar atau

video, dengan informasi yang relevan yang akan digunakan untuk melatih model. Proses ini yaitu melabelkan atau membuat tanda dengan objek yang akan kita deteksi pada kali ini kita adkan melabelkan mata kanan, mata kiri, dan mulut yang akan kita deteksi seperti Gambar 4.5 di bawah ini.

Gambar 4.5 Proses Labelling Pada Dataset

Misalnya, dalam sebuah dataset gambar, setiap gambar dapat diberi label yang menunjukkan objek atau kelas yang terkandung di dalamnya. Proses ini bisa dilakukan secara manual oleh manusia atau otomatis dengan bantuan alat dan teknologi tertentu. Labeling manual memerlukan waktu dan usaha, terutama untuk dataset besar, tetapi sering kali memberikan label yang lebih akurat. Di sisi lain, alat labeling otomatis, yang mungkin menggunakan model pembelajaran mesin sebelumnya atau algoritma deteksi objek, dapat mempercepat proses tetapi mungkin memerlukan peninjauan manusia untuk memastikan akurasi. Label yang benar dan konsisten sangat penting karena kualitas data berlabel sangat mempengaruhi kinerja dan akurasi model pembelajaran mesin yang dilatih menggunakan data seperti pada Gambar 4.6 *class* di bawah ini.



Gambar 4.6 Class Pada Dataset

4.2.3 Spliting Data

Splitting data adalah proses membagi dataset yang ada menjadi beberapa subset yang berbeda untuk tujuan pelatihan, validasi, dan pengujian model pembelajaran mesin. Proses ini penting untuk mengevaluasi kinerja model dan memastikan bahwa model tersebut dapat menggeneralisasi dengan baik pada data baru yang tidak terlihat selama pelatihan.internalnya sehingga mampu membuat prediksi atau keputusan berdasarkan data baru yang belum pernah dilihat sebelumnya *Spliting Data* terbagi menjadi 3 bagian yaitu *Train Data, Valid Data, dan Test Data*

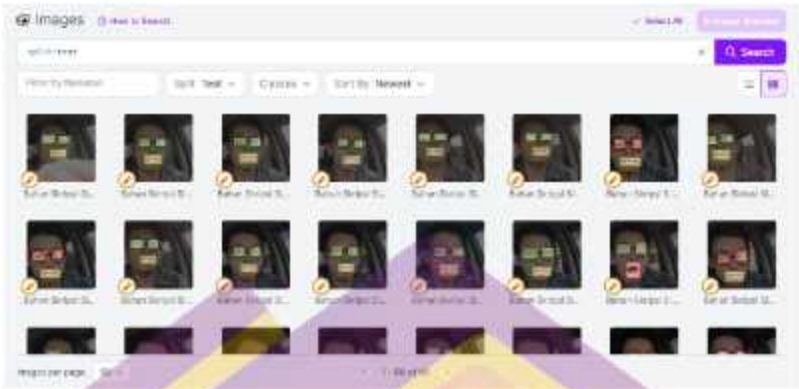
Gambar 4.7 Train Data

Pada Gambar 4.7 diatas *Train data*, atau data pelatihan, adalah sekumpulan data yang digunakan untuk melatih model pembelajaran mesin (machine learning).

Data ini berfungsi sebagai input yang digunakan model untuk mempelajari pola dan hubungan dalam data. Dengan menggunakan data pelatihan, model dapat menyesuaikan parameter internalnya sehingga mampu membuat prediksi atau keputusan berdasarkan data baru yang belum pernah dilihat sebelumnya.

Gambar 4.8 Valid Data

Pada Gambar 4.8 diatas *Valid Data*, dalam konteks pembelajaran mesin dan analisis data, merujuk pada data yang telah diverifikasi dan dipastikan keakuratannya, kebenarannya, dan kesesuaiannya untuk digunakan dalam pelatihan, validasi, dan pengujian model. Valid data adalah data yang bebas dari kesalahan, anomali, dan inkonsistensi yang dapat mengganggu kinerja model. Dengan valid data, analisis dan model pembelajaran mesin dapat menghasilkan hasil yang lebih akurat dan dapat diandalkan, serta membantu dalam pengambilan keputusan yang lebih baik.



Gambar 4.9 Split Test

Pada Gambar 4.9 diatasTest set adalah subset dari dataset yang digunakan untuk mengevaluasi kinerja akhir dari model pembelajaran mesin setelah proses pelatihan dan validasi selesai. Test set terdiri dari data yang tidak pernah dilihat oleh model selama pelatihan dan validasi, sehingga memberikan gambaran yang objektif tentang bagaimana model akan berkinerja pada data baru yang tidak dikenal. Dengan menggunakan test set, kita dapat menilai secara objektif kinerja model dan memastikan bahwa model yang dibangun memiliki kemampuan generalisasi yang baik dan dapat diandalkan untuk aplikasi nyata.

Hasil *Training Set, Validation Set,* dan *Testing Set* bisa dilihat dari tabel 4.0 dan 4.1 dibawah. ini terbagi menajdi 2 hasil dari dataset siang dan malam hari.

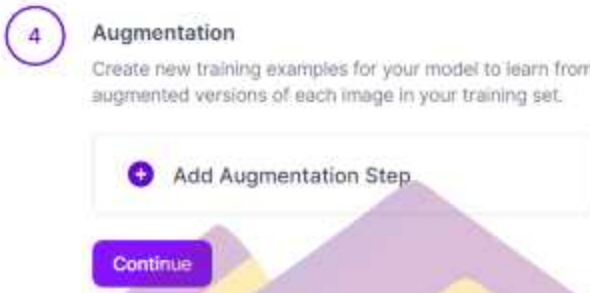
Tabel 4.0 Hasil *Train Set, Valid Set*, dan *Test Set* Dataset Siang Hari Hari.

Training Set	***Validation Set***	***Testing Set***
417	119	60

Tabel 4.1 Hasil Train Set, Valid Set, dan Test Set Dataset Malam Hari.

Training Set	***Validation Set***	***Testing Set***
335	96	48

4.2.4 Data Augmentation



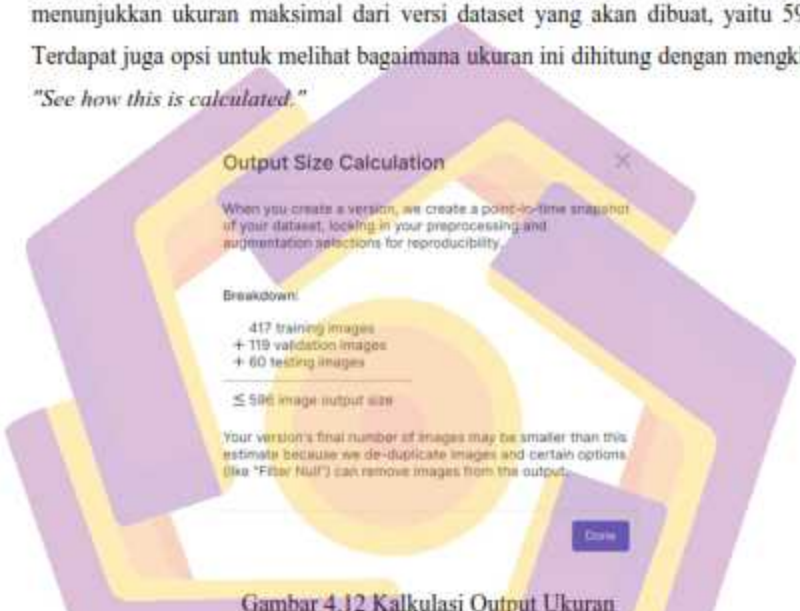
Gambar 4.10 Augmentasi Data

Pada Gambar 4.10 Augmentasi data adalah teknik yang digunakan dalam machine learning, khususnya dalam bidang computer vision, untuk meningkatkan ukuran dan keragaman dataset tanpa perlu mengumpulkan data tambahan. Proses ini melibatkan pembuatan variasi dari dataset yang ada melalui berbagai transformasi, seperti rotasi, pergeseran, perubahan ukuran, flipping, penambahan noise, dan penyesuaian kecerahan atau kontras. Dengan menerapkan augmentasi data, model dapat lebih baik menangkap berbagai variasi dan generalisasi dalam data, sehingga meningkatkan kinerja dan akurasinya.

Gambar 4.11 Proses Generate Project

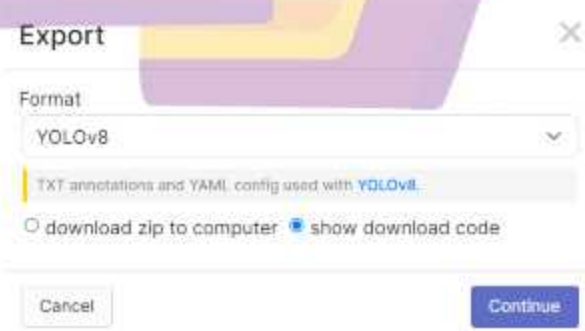
Pada Gambar 4.11 diatas tersebut menunjukkan langkah kelima dalam proses menggunakan Roboflow, yaitu langkah *"Create."* Pada langkah ini, pengguna

diminta untuk meninjau pilihan yang telah mereka buat sebelumnya, seperti penyesuaian data atau langkah-langkah prapemrosesan. Setelah peninjauan, pengguna dapat mengklik tombol *"Create"* untuk membuat versi snapshot dataset mereka, yang mencakup semua langkah prapemrosesan yang telah diterapkan. Pada Gambar 4.12 dibawah juga menyebutkan "Maximum Version Size" yang menunjukkan ukuran maksimal dari versi dataset yang akan dibuat, yaitu 596. Terdapat juga opsi untuk melihat bagaimana ukuran ini dihitung dengan mengklik *"See how this is calculated."*



Gambar 4.12 Kalkulasi Output Ukuran

4.2.5 Export Dataset



Gambar 4.13 Export Dataset

Pada Gambar 4.13 diatas tersebut menunjukkan antarmuka "Export" pada platform Roboflow, di mana pengguna dapat memilih format keluaran untuk dataset mereka. Format yang dipilih dalam gambar ini adalah "YOLOv8," yang merupakan versi terbaru dari model *YOLO (You Only Look Once)* yang digunakan untuk deteksi objek. Gambar juga menunjukkan bahwa anotasi dalam format TXT dan file konfigurasi YAML akan disertakan sesuai dengan standar YOLOv8. Untuk melanjutkan, pengguna dapat mengklik tombol "Continue" di bagian kanan bawah.

Gambar 4.14 Kode API Dataset

Pada Gambar 4.14 diatas tersebut menunjukkan sebuah kotak dialog yang berisi kode untuk mengunduh dataset dari Roboflow menggunakan Python. Terdapat tiga pilihan untuk mengakses kode tersebut: melalui Jupyter Notebook, Terminal, atau URL mentah (Raw URL). Kode ini termasuk perintah untuk menginstal pustaka Roboflow (!pip install roboflow) dan untuk mengimpor pustaka tersebut dalam skrip Python. Terdapat peringatan untuk tidak membagikan potongan kode ini di luar tim pengguna, karena kunci API yang digunakan bersifat pribadi dan terkait dengan akun Roboflow pengguna

4.2.6 Training Data

Fungsi utama dari training data pada *Google Colaboratory* adalah untuk melatih model pembelajaran mesin menggunakan dataset yang disediakan dalam

lingkungan cloud yang efisien dan mudah digunakan. *Google Colaboratory* menyediakan platform yang lengkap untuk menulis dan menjalankan kode Python, mendukung berbagai pustaka pembelajaran mesin seperti TensorFlow, Keras, PyTorch, dan Scikit-Learn. Dengan *Google Colaboratory*, pengguna dapat melakukan eksperimen dan penelitian tanpa keterbatasan perangkat keras lokal, serta memanfaatkan akses gratis ke GPU dan TPU untuk mempercepat proses pelatihan model. Selain itu, Colab menyediakan alat visualisasi yang kuat, seperti Matplotlib dan TensorBoard, untuk memantau kinerja model selama pelatihan, termasuk metrik seperti akurasi dan loss. Fitur penyimpanan dan berbagi di *Google Colaboratory* memungkinkan pengguna untuk menyimpan model terlatih, dataset, dan hasil eksperimen di *Google Drive* atau *Google Cloud Storage*, serta berbagi notebook dengan rekan kerja atau komunitas. Secara keseluruhan, *Google Colaboratory* memfasilitasi proses pelatihan model pembelajaran mesin dengan menyediakan alat yang diperlukan untuk memuat, memproses, dan melatih data secara efisien.

4.2.7 Install Yolo Pada Google Colaboratory

Selanjutnya yaitu kita menginstall *YOLO* dengan memasukan code seperti pada Code 4.1 di bawah ini.

```
# !pip install method (recommended)

!pip install ultralytics==8.0.196

from IPython import display
display.clear_output()

import ultralytics
ultralytics.checks()
```

Code 4.1 Install *YOLO* Pada *Google Colaboratory*.

Setelah berhasil menginstall yolo Langkah selanjutnya yaitu mengimpor yolo seperti pada code 4.2 di bawah ini.

```
from ultralytics import YOLO

from IPython.display import display, Image
```

Code 4.2 Import YOLO.

Proses Import Yolov8 untuk menjadikan dataset sebagai bahan untuk diolah.

4.2.8 Testing YOLOv8

Langkah selanjutnya kita mengetest yolo menggunakan gambar anjing seperti Code 4.3 di bawah ini.

```
%cd {HOME}
!yolo task=detect mode=predict model=yolov8n.pt conf=0.25
source='https://media.roboflow.com/notebooks/examples/dog.jpeg
' save=True
```

```
%cd {HOME}
Image(filename='runs/detect/predict/dog.jpeg', height=600)
```

Code 4.3 Testing YOLO

4.3 Model Testing

Model testing pada *Google Colaboratory* merujuk pada proses pengujian dan evaluasi kinerja model machine learning atau deep learning yang dijalankan menggunakan infrastruktur *Google Colaboratory*. Ini melibatkan penggunaan sumber daya komputasi yang kuat yang disediakan secara gratis oleh Google, termasuk akses ke GPU dan TPU. Proses model testing ini umumnya mencakup langkah-langkah yaitu Implementasi Model, Persiapan Data, Tokenisasi dan *Preprocessing* (Opsional), Inferensi dan Evaluasi, dan Visualisasi Hasil (Opsional).

4.3.1 Proses Download dan Import Dataset Dari Roboflow

```
!mkdir {HOME}/datasets
%cd {HOME}/datasets

!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="s1nWSa2gDFA5CPPGRRfm")
project = rf.workspace("arif-farhan-mahardika-
putra").project("deteksi-mata-kantuk-siang-hari")
version = project.version(1)
```

```
dataset = version.download("yolov8")
```

Code 4.4 Source Code Download dan Import Dataset Roboflow

Pada Code 4.3 Proses *download* dan *import* dataset yang berasal dari *Roboflow* ke *Google Colaboratory* menggunakan API yang tertera pada Gambar 4.4 diatas.

4.3.2 Clone Training Notebook Yolov8

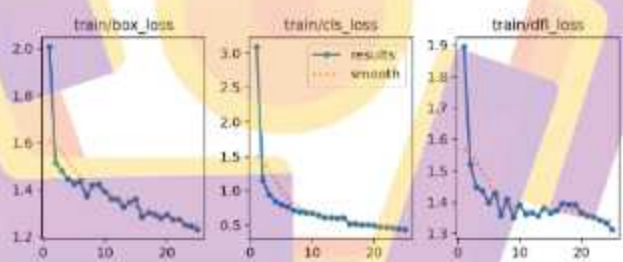
```
%cd {HOME}

!yolo task=detect mode=train model=yolov8s.pt
data={dataset.location}/data.yaml epochs=25 imgsz=800
plots=True
```

Code 4.5 Proses Clone Yolov8 dari Github

Pada Code 4.5 diatas tahap ini peneliti menulis code untuk cloning repository *YOLOv8* dari *Github* ke *Google Colaboratory*. Pada proses ini, peneliti menggunakan epochs sebanyak 25 dan *imagesize* nya sebesar 800.

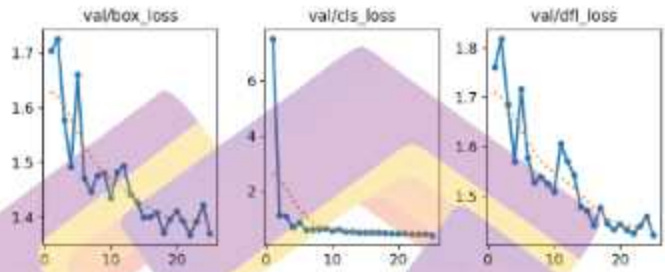
4.4 Evaluasi Model



Gambar 4.15 Hasil Dari Pengujian Train Model Testing

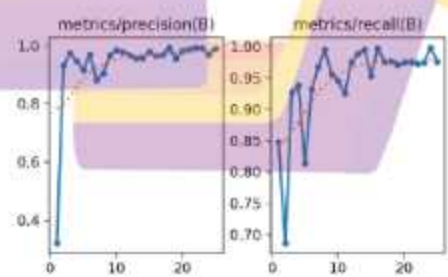
Berdasarkan data yang telah didapatkan pada Gambar 4.15 diatas terlihat hasil *training box_loss* yang turun dari *epoch* pertama yaitu 2,00 langsung turun ke 1,50 lalu terjadi penurunan pada *epoch* 1,50 sampai ke *epoch* selanjutnya dengan nilai 1,58 lalu mengalami nilai *epoch* yang naik turun sampai ke nilai 1,45 selepas itu turun hingga *epoch* terakhir. Lalu selanjutnya pada *training cls_loss* terjadi penurunan yang drastis pada *epoch* pertama yaitu 3,30 langsung turun ke 1,20 lalu terjadi penurunan bertahap ke *epoch* terakhir dengan nilai 0,46. Pada *training df_loss* terjadi penurunan pada *epoch* pertama dengan nilai 1,90 langsung turun

signifikan ke 1,52 lalu turun hingga ke *epoch* selanjutnya dengan nilai 1,44 lalu menurun sedikit dan naik turun dari *epoch* sebelum-nya ke *epoch* terakhir dengan nilai 1,30 di tengah penurunan nilai ada peningkatan nilai dan penurunan lagi hingga *epoch* terakhir.



Gambar 4.16 Hasil *Evaluation* Dari Pengujian *Validation Model Testing*

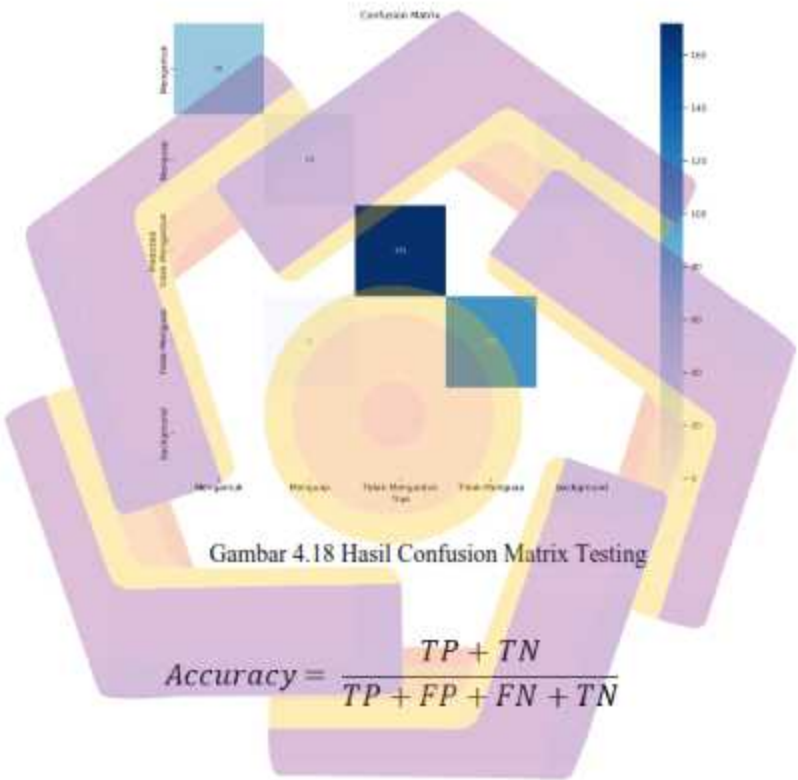
Pada Gambar 4.16 diatas terlihat pada hasil *validation box_loss* mengalami grafik yang fluktuatif mengalami penurunan dan peningkatan hingga epoch akhir. Pada *validation cls_loss* mengalami penurunan drastic yang kedua setelah kedua baru mengalami penurunan yang tidak signifikan hingga akhir. Pada *validation dfl_loss* terjadi *epoch* yang fluktuatif sempat naik pada awal lalu turun lagi lalu turun drastis pada pertengahan sempat naik lagi dan sempat turun kembali lalu menurun terus hingga akhir.



Gambar 4.17 Hasil Kurva Nilai Precision dan Nilai Recall

Dapat dilihat pada Gambar 4.17 diatas bahwa kurva nilai *precision* mengalami peningkatan yang sangat pesat pada awal lalu mengalami grafik yang

fluktuatif dan pada akhir kurva terjadi kenaikan hingga ke nilai *epoch* 0,98. Pada kurva nilai recall terlihat terjadi penurunan secara drastis hingga mencapai titik tertinggi dengan nilai 0,99 lalu mengalami penurunan dan mengalami peningkatan lagi lalu terjadi kesatabilan diakhir dan sampai akhir terjadi kenaikan dan penurunan lagi.



Gambar 4.18 Hasil Confusion Matrix Testing

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Gambar 4.19 Rumus Penghitungan Accuracy

Accuracy = TP + TN = 172 + 107 = 279

= TP + TN + FP + FN = 172 + 107 + 1 + 1 = 281

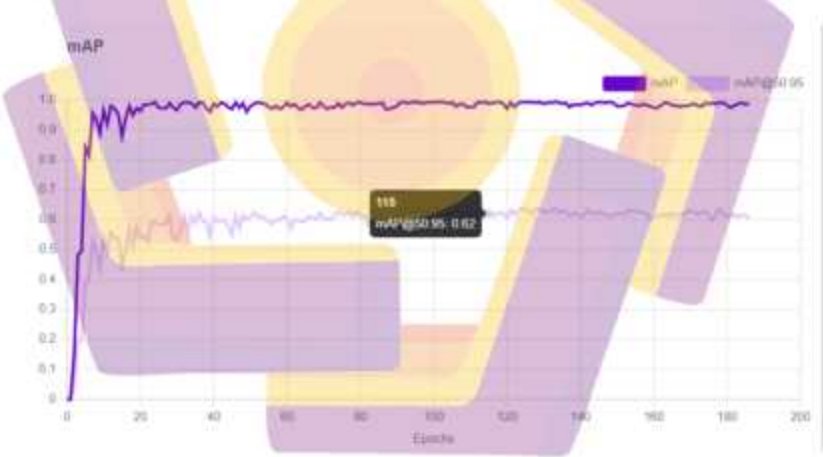
Accuracy = 279 ÷ 281 = 0.99288

= 0.99288 x 100% = 99,28%

Accuracy yang diperoleh dari proses proses pengujian ini adalah 0.99288 atau 99,28% dengan menghitung dari Confusion Matrix pada Gambar 4.19 diatas.

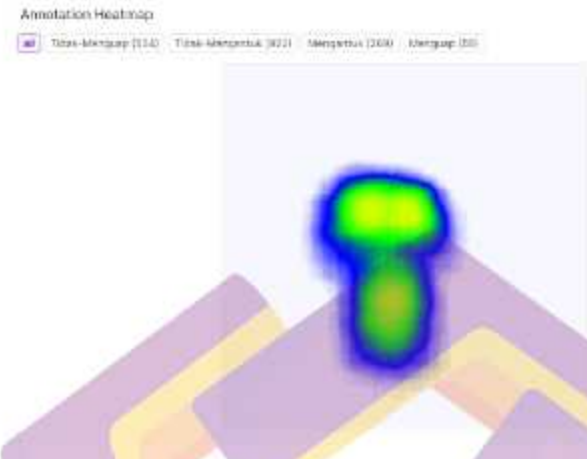
Gambar 4.20 Detail Dataset pada Roboflow

Pada Gambar 4.20 diatas bahwa dataset yang dipakai pada penelitian ini memiliki *mAP* sebesar 99,3 % lalu memiliki presentase *Precision* sebesar 99,0% lalu memiliki presentase *Recall* sebesar 97,5%



Gambar 4.21 Grafik mAP pada Roboflow

Terlihat pada grafik *mAP* pada Gambar 4.21 diatas bahwa *mAP* mengalami peningkatan yang sangat pesat dari nilai 0 ke 0,85. Selanjutnya grafik mengalami nilai yang naik turun tetapi masih stabil hingga ke *epoch* terakhir.



Gambar 4.22 Hasil Annotation Heatmap Dataset pada Roboflow

Terlihat pada *Annotation Heatmap* pada Gambar 4.22 diatas memiliki tingkat kepadatan yang berbeda beda tergantung pada warna yang terlampir pada *heatmap*

Gambar 4.23 Hasil Generate Dataset pada Roboflow

Hasil *generate* dataset yang berada di *Roboflow* pada Gambar 4.23 diatas menghasilkan sebanyak 596 total gambar lalu terdapat 1,780 *annotations* atau *labelling* pada 4 *class* lalu memiliki *average image size* sebesar 2,07 mp dan memiliki median *image ratio* berukuran 1920x1080. Selain itu, terlihat juga nilai class balance dari 4 total *class* yang ada di dataset.

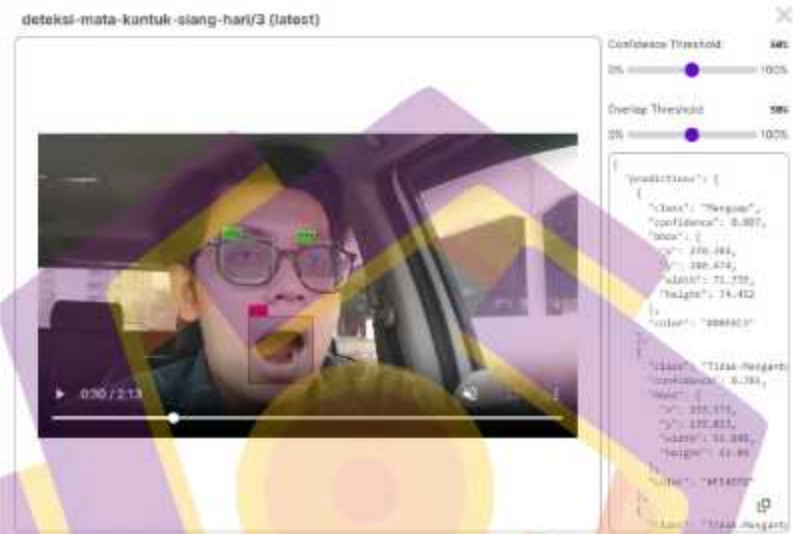
4.5 Interpertasi Hasil

Gambar 4.24 Hasil Generate Dataset Keaadan Terjaga pada Roboflow.

Pada gambar 4.24 diatas terlihat bahwa proses penelitian ini telah berhasil mendeteksi kedua mata dan mulut dalam keadaan tetap terjaga berwarna hijau semua.

Gambar 4.25 Hasil Generate Dataset Keaadan Mengantuk pada Roboflow.

Pada Gambar 4.25 diatas terlihat bahwa proses penelitian ini telah berhasil mendeteksi kedua mata dan mulut dalam keadaan mata mengantuk dia akan berubah berwarna merah dan mulut tetap terjaga berwarna hijau.



Gambar 4.26 Hasil Generate Dataset Dalam Keaadan Terjaga dan Mneguap pada Roboflow.

Pada Gambar 4.26 diatas terlihat bahwa proses penelitian ini telah berhasil mendeteksi kedua mata dan mulut dalam keadaan tetap terjaga berwarna hijau, dan mulut menguap berwarna merah.

Tabel 4.2 Hasil Pengujian Dataset Dari Berbagai Kondisi

Nama Dataset	Detail Dataset	Hasil mAP	Hasil Precision	Hasil Recall	Hasil Accuracy
Deteksi Mata Kantuk Pada Siang Hari (Menggunakan Kacamata)	417 Images Train Set 119 Images Valid Set 60 Images Test Set Total = 596	99,3%	99,0%	97,5%	99,28%
Deteksi Mata Kantuk Pada Malam Hari(Menggunakan Kacamata)	335 Images Train Set 98 Images Valid Set 48 Images Test Set Total = 481	97,8%	92,8%	94,8%	99,07%
Deteksi Mata Kantuk Siang Hari Berbagai Orang (Tanpa Kacamata)	226 Images Train Set 65 Images Valid Set 32 Images Test Set Total = 323	98,8%	96,7%	99,0%	98,54%

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan tabel hasil penelitian deteksi mata mengantuk dengan arsitektur YOLO, beberapa kesimpulan dapat diambil. Penelitian ini mengevaluasi deteksi mata kantuk pada siang dan malam hari dengan menggunakan kacamata, serta deteksi pada berbagai orang tanpa kacamata. Pada siang hari dengan kacamata, model dilatih menggunakan 417 gambar, divalidasi dengan 119 gambar, dan diuji dengan 60 gambar, total sebanyak 596 gambar, dengan akurasi masing-masing sebesar 99,3% untuk set pelatihan, 99,0% untuk set validasi, dan 97,5% untuk set pengujian, serta akurasi keseluruhan sebesar 99,28%. Pada malam hari dengan kacamata, model dilatih menggunakan 335 gambar, divalidasi dengan 98 gambar, dan diuji dengan 48 gambar, total sebanyak 481 gambar, dengan akurasi masing-masing sebesar 97,8% untuk set pelatihan, 92,8% untuk set validasi, dan 94,8% untuk set pengujian, serta akurasi keseluruhan sebesar 99,07%. Untuk deteksi mata kantuk siang hari tanpa kacamata pada berbagai orang, model dilatih menggunakan 226 gambar, divalidasi dengan 65 gambar, dan diuji dengan 32 gambar, total sebanyak 323 gambar, dengan akurasi masing-masing sebesar 98,8% untuk set pelatihan, 96,7% untuk set validasi, dan 99,0% untuk set pengujian, serta akurasi keseluruhan sebesar 98,54%. Secara keseluruhan, sistem deteksi mata mengantuk dengan arsitektur YOLO menunjukkan performa yang sangat baik dalam berbagai kondisi, baik pada siang hari maupun malam hari, serta dengan atau tanpa kacamata, dengan akurasi yang konsisten tinggi di semua skenario pengujian. Berdasarkan penelitian yang dilakukan, dapat disimpulkan bahwa arsitektur YOLOv8 dapat diimplementasikan secara efisien dalam sistem deteksi mata kantuk dengan menghasilkan hasil yang akurat dan cepat. Sistem ini menunjukkan performa yang baik dalam berbagai kondisi pencahayaan dan latar belakang visual, meskipun ada variasi dalam tingkat akurasi deteksi tergantung pada kondisi lingkungan. Selain itu, analisis terhadap subyek berkacamata dan tidak berkacamata menunjukkan bahwa penggunaan kacamata dapat mempengaruhi kinerja deteksi, dengan perbedaan kecil dalam tingkat akurasi. Hasil ini

menunjukkan bahwa sistem perlu dipertimbangkan untuk beradaptasi dengan aksesori wajah guna meningkatkan akurasi deteksi dalam semua kondisi.

5.2 Saran

Untuk penelitian tentang pengaruh deteksi mata mengantuk menggunakan kacamata atau tidak berkacamata dengan metode arsitektur YOLO, beberapa saran yang dapat membantu meningkatkan kualitas penelitian meliputi:

1. Kembangkan Aplikasi Praktis
   Pertimbangkan bagaimana hasil penelitian dapat diterapkan dalam sistem peringatan tidur untuk pengemudi atau operator mesin untuk meningkatkan keselamatan
2. Uji di Dunia Nyata
   Lakukan pengujian lapangan untuk memastikan bahwa model bekerja dengan baik dalam kondisi dunia nyata, bukan hanya dalam lingkungan yang dikendalikan.
3. Peningkatan Akurasi Deteksi
   Pertimbangkan untuk menyertakan model deteksi wajah atau mata sebelumnya sebagai preprocessing sebelum YOLO, untuk meningkatkan akurasi deteksi mata secara khusus.
4. Penyesuaian Model YOLO
   Uji berbagai versi dari arsitektur YOLO (misalnya YOLOv3, YOLOv4, atau YOLOv5) untuk menemukan yang paling efektif dalam mendeteksi mata mengantuk dengan dan tanpa kacamata.

REFERENSI

[1] P. B. Reddy and S. Rachana, *DRIVER DROWSINESS MONITORING*, vol. 11, no. 6 June, 2023.

[2] E. U. Armin, A. P. Edra, F. I. Alifin, I. Sadidan, I. P. Sary dan L. Ulinnuha, *Performa Model YOLOv8 untuk Deteksi Kondisi Mengantuk*, 2023.

[3] M. N. I. Muhlasin and A. Stefanie, *KLASIFIKASI PENYAKIT MATA BERDASARKAN CITRA FUNDUS*, vol. 7, 2023.

[4] Y. F. Aziz and I. , *YOLO-V8 PENINGKATAN ALGORITMA UNTUK DETEKSI PEMAKAIAN MASKER WAJAH*, vol. 7, 2023.

[5] H. A. Fauzan and K. Ari, *Aplikasi Warning Alert Pendeteksi Kelelahan Ekspresi Wajah Pada Pengemudi Secara Real-Time Menggunakan Metode You Only Look Once Berbasis Website*, 2023.

[6] I. M. Ayub, "Apa Itu AI (Artificial Intelligence): Pengertian, Kelebihan, Dan Kekurangan," 12 Juli 2023. [Online]. Available: https://stekom.ac.id/artikel/apa-itu-ai-kecerdasan-buatan-pengertian-kelebihan-kekurangan.

[7] I. Salamah, M. R. A. Ali Said and S. Soim, *Perancangan Alat Identifikasi Wajah Dengan Algoritma You Only Look Once (YOLO) Untuk Presensi Mahasiswa*, vol. 6, 2022.

[8] P. dr., "Kantuk," 14 April 2022. [Online]. Available: https://www.alodokter.com/kantuk.

[9] F. Dita, "Pengertian Machine Learning Model & Implementasinya," 23 Januari 2023. [Online]. Available: https://dqlab.id/pengertian-machine-learning-model-and-implementasinya.

[10] P. Rani, "Deep Learning: Model AI yang Terinspirasi dari Otak Manusia," 16 Desember 2023. [Online]. Available: https://phintraco.com/deep-learning/.

[11] H. Gagan, "Apa itu Computer Vision: Penggunaan dan Manfaatnya," 20 Oktober 2023. [Online]. Available: https://metanesia.id/blog/apa-itu-computer-vision.

[12] S. Haziqa, "Membongkar Yolov8: Mahakarya Visi Komputer Viral Ultralytics," 13 Januari 2024. [Online]. Available: https://www.unite.ai/id/ultralitik-yolov8-menjelaskan/.

[13] R. J and F. A., *YOLOv3: An Incremental Improvement.*, 2023.

[14] F. Ciaglia, *Roboflow 100: A Rich, Multi-Domain Object Detection Benchmark*, 2022.

[15] F. M. Talaat and Z. Hanaa, *An improved fire detection approach based on YOLO-v8 for smart*, 2023.

[16] M. Talib, A. H. Y. Al-Noori and J. Su, *Improved YOLOv8 for Real-time object detection*, vol. 10, no. 1, 2024.

[17] X. Wang, H. Gao, Z. Jia and Z. Li, *BL-YOLOv8: An Improved Road Defect Detection Model Based*, 2023.

LAMPIRAN

Lampiran 1 Install YOLOv8

```
# Pip install method (recommended)

!pip install ultralytics==8.0.196

from IPython import display
display.clear_output()

import ultralytics
ultralytics.checks()
```

Lampiran 2 Import YOLOv8

```
from ultralytics import YOLO

from IPython.display import display, Image
```

Lampiran 3 Export Dataset

```
!mkdir {HOME}/datasets
%cd {HOME}/datasets

!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="sinWSa2gDPASCPPGRRfm")
```

```
project = rf.workspace("arif-farhan-mahardika-
putra").project("deteksi-mata-kantuk-siang-hari")
version = project.version(3)
dataset = version.download("yolov8")
```

Lampiran 4 CUSTOM TRAINING

```
%cd {HOME}

!yolo task=detect mode=train model=yolov8s.pt
data={dataset.location}/data.yaml epochs=25 imgsz=800
plots=True
```

```
!ls {HOME}/runs/detect/train/
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/confusion_matrix.png
', width=600)
```

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/results.png',
width=600)
```

```
%cd {HOME}
Image(filename=f'{HOME}/runs/detect/train/val_batch0_pred.jpg'
, width=600)
```

Lampiran 5 VALIDATE CUSTOM MODEL

```
%cd {HOME}

!yolo task=detect mode=val
model={HOME}/runs/detect/train/weights/best.pt
data={dataset.location}/data.yaml
```

Lampiran 6 INFRENCE WITH CUSTOM MODEL

```
%cd {HOME}
!yolo task=detect mode=predict
model={HOME}/runs/detect/train/weights/best.pt conf=0.25
source={dataset.location}/test/images save=True
```

```
import glob
from IPython.display import Image, display

for image_path in
glob.glob(f'{HOME}/runs/detect/predict3/*.jpg')[:3]:
      display(Image(filename=image_path, width=600))
      print("\n")
```

Lampiran 7 DEPLOY MODEL ON ROBOFLOW

```
project.version(dataset.version).deploy(model_type="yolov8",
model_path=f"{HOME}/runs/detect/train/")
```

```
#Run inference on your model on a persistant, auto-scaling,
cloud API

#load model
model = project.version(dataset.version).model

#choose random test set image
import os, random
test_set_loc = dataset.location + "/path/to/your/image.jpg"
random_test_image = random.choice(os.listdir(test_set_loc))
print("running inference on " + random_test_image)

pred = model.predict(test_set_loc + random_test_image,
confidence=40, overlap=30).json()
pred
```